2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology

Multi-Concept Document Classification Using A Perceptron-like Algorithm

Clay Woolam
University of Texas at Dallas
cpw021000@utdallas.edu

Latifur Khan
University of Texas at Dallas
lkhan@utdallas.edu

Abstract

Previous work in hierarchical categorization focuses on the hierarchical perceptron (Hieron) algorithm. Hierarchical perceptron works on the principles of the perceptron, that is each class label in the hierarchy has an associated weight vector. To account for the hierarchy, we begin at the root of the tree and sum all weights to the target label. We make a prediction by considering the label that yields the maximum inner product of its feature set with its path-summed weights. Learning is done by adjusting the weights along the path from the predicted node to the correct node by a specific loss function that adheres to the large margin principal. There are several problems with applying this approach to a multiple class problem. In many cases we could end up punishing weights that gave a correct prediction, because the algorithm can only take a single case at a time. In this paper we present an extended hierarchical perceptron algorithm capable of solving the multiple categorization problem (MultiHieron). We introduce a new aggregate loss function for multiple label learning. We make weight updates simultaneously instead of serially. Then, significant improvement over the basic Hieron algorithm is demonstrated on the Aviation Safety Reporting System (ASRS) flight anomaly database and OntoNews corpus using both flat and hierarchical categorization metrics.

1 Introduction

The emerging semantic web will provide the most significant information resource ever available to humanity. Consequently, it happens to be one of the most daunting engineering tasks of all time. We are investigating an algorithm that may be a step in the development of a major semantic web area, *metadata generation*.

An ontology is a specification of an abstract, simplified view of the world that we wish to represent for some purpose [3]. Therefore, an ontology defines a set of representational terms that we call concepts. Interrelationships among these concepts describe a target world. Metadata generation is a process of extracting structured machine-understandable information from documents. Ontology based metadata extraction (OBME) aims to automatically extract metadata from text to concepts in an ontology. OBME is a critical part of building a useful semantic web.

In this paper, we further investigate a large margin hierarchical perceptron algorithm developed by Dekel et al. [1] in 2004. It was shown to produce good results at entire document *classification* when document class labels were hierarchical in nature. Later, in 2007, Yaoyong and Bontcheva [4] demonstrated promising results by applying this algorithm to OBME in 2007. Previous work with this algorithm focused only on *single class labeling*. It has not been shown to work well if more than one class label exists. We extend this to do multiple label anomaly classification by performing *simultaneous* updates.

The paper is organized in the following way. Section 2 of this paper describes the traditional Hieron algorithm as has been explored in previous work. Section 3 describes a new, more general algorithm, capable of learning multiple concepts at a time for one document. Section 4 explores the generality of the new algorithm while establishing the same bounds as the traditional Hieron algorithm. Section 5 discusses the database and provides evidence that learning with the new algorithm improves on learning in the single document case. Section 6 discusses conclusions and explores future possibilities for this research.

2 Hierarchical perceptron algorithm

The hierarchical perceptron algorithm is named Hieron. It was originally presented as a fully developed Online Hieron algoirthm, which we will briefly go over here.

To describe the algorithm, we first establish some definitions. Labels are hierarchical and each has a corresponding weight that is a vector of real numbers, $\mathbf{w}^v \in \mathbb{R}^n$ where $v \in Y$. Notice how we superscript the weight vector with the label name to refer to the weight corresponding to that label. Also, we call Y our set of all class labels. To refer to a set of all class labels from the top of the hierarchy to a given label we say $P(v)$. We classify a vector of real num-

978-0-7695-3496-1/08 $25.00 © 2008 IEEE
DOI 10.1109/WIIAT.2008.410

bers called x. Finally, predictions are made according to the rule in equation 1. For every possible label, we sum all of the weights from the top of the hierarchy to that label, then we multiply it by our instance. The largest summation is the prediction.

$$f(x) = \arg\max_{v \in Y} \sum_{u \in P(v)} \mathbf{w}^u \cdot \mathbf{x} \quad (1)$$

Next, we need to train the weights in the system. Initially they are set to zero. At each timestep i, we query the system for a prediction $\hat{y}_i$ for the instance x_i. The error is the *tree distance*, or the shortest path from a predicted label to its corresponding true label, denoted by $\gamma(y_i, \hat{y}_i)$. The algorithm then assumes that there exists a set of weights $\{\omega^v\}_{v \in y}$ such that for all instance-label pairs ($\mathbf{x}_i$,y_i) in the training set and $r \in Y \backslash \{y_i\}$, the following holds

$$\sum_{v \in P(y_i)} \omega^v \cdot \mathbf{x}_i - \sum_{u \in P(r)} \omega^u \cdot \mathbf{x}_i \geq \sqrt{\gamma(y_i, r)} \quad (2)$$

From this, we have a loss function that forms the basis for our update,

$$L(\{\mathbf{w}^v\}, \mathbf{x}_i, y_i) = \sum_{v \in P(y_i)} \omega^v \cdot \mathbf{x}_i - \sum_{u \in P(r)} \omega^u \cdot \mathbf{x}_i + \sqrt{\gamma(y_i, r)} \quad (3)$$

Algorithm 1 Online Hieron

Initialize: $\forall v \in Y : \mathbf{w}_0^v = \mathbf{0}, i = 0$
1. **for** $i = 1$ **to** m
Using $(\mathbf{x}_i, \mathbf{y}_i)$
2. Predict: $\hat{y} = \arg\max_{y \in Y} \sum_{v \in P(y)} \mathbf{w}_i^v \cdot \mathbf{x}_i$
3. Update:
$w_{i+1}^v = w_i^v + \alpha_i \mathbf{x}_i$**, if** $v \in P(y) \backslash P(\hat{y})$
$w_{i+1}^v = w_i^v - \alpha_i \mathbf{x}_i$**, if** $v \in P(\hat{y}) \backslash P(y)$
where $\alpha_i = \frac{L(\{\mathbf{w_i^v}\}, \mathbf{x}_i, y_i)}{\gamma(y_i, \hat{y}) ||\mathbf{x}_i||^2}$

Algorithm 1 is the basic Online Hieron algorithm that has been subject to formal analysis. In line 1, we iterate through our entire training set, which has m training instances. On each iteration we have a new $(\mathbf{x}_i, \mathbf{y}_i)$. In line 2, a prediction is made. If it's correct, no update is made on line 3, if false, then only relevant nodes are updated. Updates are made using a update rule derived by Dekel et al.[1] This update rule uses a loss function from equation 3. Using lagrangian multipliers we find the optimal update policy, denoted above by alpha.

3 Multi-category hierarchical perceptron

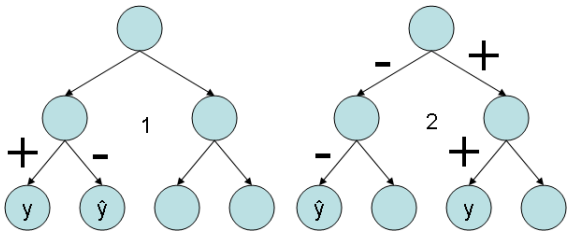
As mentioned, the Hieron algorithm has shown promising results for the single label categorization problem. There are many cases, especially with regards to ontologies, where we need to do multi-category prediction. The same prediction model still holds, the prediction simply need be refined to include the highest n resulting path-instance products.

Unfortunately, learning using the ordinary Hieron algorithm had a few problems. Initially we use learning algorithm 1, however, intuition tells us that there are a few problems with this method. First, the problem that the original algorithm was attempting to solve was framed in terms of one prediction label for one true label, not many labels, so our update rule may not work well. Second, we could have a case where we erase important information when updates are made for different labels on the same document as the following example shows. Imagine we have a document that has 2 labels. Depicted in figure 1 is such a document following two different weight update rules. Figure 1(a), shows how we may have an undesirable update case by repeating predictions. The bottom right node represents the correct label (y) and its sibling represents the predicted node ($\hat{y}$). This is incorrect, so the weights on the true node will be rewarded (denoted by $+$ in figure 1(a)) while the weights on the prediction will be punished (denoted by $-$ in figure 1(a)). Because they share the same parent, no other nodes need be updated. When queried for another prediction, it makes what would be the correct label in step 1, but is the incorrect label for step 2. Here, the left-most leaf node and second right most leaf node are predicted and the true class, respectively (tree in figure 1(a)). Updating now cancels out half of the information we added to the system in step 1. It is also expensive, 6 nodes have been updated in the process of training one document.

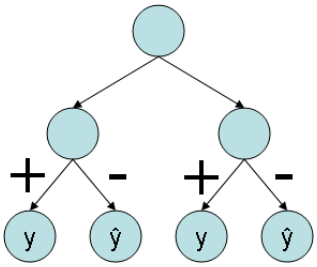
In figure 1(b), we propose a simulataneous update strategy. Only 4 updates are made, on each of leaf node in the tree. Both predictions are incorrect, but appear as siblings of the true values, therefore only those nodes need be adjusted and the rest of the tree remains untouched. In other words, left most leaf and second right most leaf nodes are rewarded; second most leaf and right most leaf nodes are punished. As the following sections will demonstrate, not only does this method intuitively feel better, but it is advantageous in several ways. By updating everything in one shot, we are taking into account all information available for a more complete result. By making multiple predictions at once we end up making fewer weight updates.

3.1 Proposed algorithm

To develop a better algorithm, the problem must be redefined in multi-category terms. The new algorithm will assume that, for all instance-label pairs ($\mathbf{x}_i$,$\mathbf{y}_i$) in the training set and $\mathbf{r} \subset Y$ such that at least one $r_j \in Y \backslash \mathbf{y}_i$, the



(a) Original Hieron learning.



(b) MultiHieron learning.

Figure 1. Example of learning weights for the same document containing two labels.

following margin requirements hold

$$\sum_{z\in\mathbf{y}_i}\sum_{v\in P(z)}\omega^v\cdot\mathbf{x}_i-\sum_{q\in\mathbf{r}}\sum_{u\in P(r)}\omega^u\cdot\mathbf{x}_i \geq\sqrt{\gamma(\{\mathbf{y}\},\{\mathbf{r}\})} \tag{4}$$

From here, and taking a similar approach to the method used in [1], we will craft MultiHieron, the multi-category hierarchical perceptron algorithm. First, notice that our tree distance function has changed characteristics slightly. We now denote $\gamma(\mathbf{V},\mathbf{U})$ to be

$$\gamma(\mathbf{V},\mathbf{U})=|\bigcup_{v\in\mathbf{V}}P(v)\triangle\bigcup_{u\in\mathbf{U}}P(u)| \tag{5}$$

Note that $\triangle$ denotes the symmetric difference ($A\triangle B=(A\backslash B)\bigcup(B\backslash A)$) between sets. Following this, our loss function becomes

$$L(\{\mathbf{w_i^v}\},\mathbf{x}_i,\mathbf{y}_i)=\sum_{y\in\mathbf{y}_i}\sum_{v\in P(y)}\mathbf{w}_i^v\cdot\mathbf{x}_i-\sum_{r\in\mathbf{r}}\sum_{u\in P(r)}\mathbf{w}_i^u\cdot\mathbf{x}_i+\sqrt{\gamma(\mathbf{y}_i,\mathbf{r})} \tag{6}$$

To control learning by ensuring that the margin requirement is met but also keeping the updated weights close to the previous, we follow the method provided in [1] and say,

$$\min_{\{\mathbf{w}^v\}}\frac{1}{2}\sum_{v\in Y}||\mathbf{w}^v-\mathbf{w}_i^v||^2$$
$$s.t.\sum_{y\in\mathbf{y}_i}\sum_{v\in P(y)}\mathbf{w}^v\cdot\mathbf{x}_i-\sum_{\hat{y}\in\hat{\mathbf{y}}_i}\sum_{u\in P(\hat{y})}\mathbf{w}^u\cdot\mathbf{x}_i\geq\sqrt{\gamma(\mathbf{y}_i,\hat{\mathbf{y}}_i)} \tag{7}$$

We can solve this condition using Lagrange multipliers, α_i and preform some optimization to get:

$$\alpha_i=\frac{L(\{\mathbf{w}_i^v\},\mathbf{x}_i,y_i)}{\gamma(\mathbf{y}_i,\hat{\mathbf{y}}_i)||\mathbf{x}_i||^2} \tag{8}$$

Algorithm 2 Online MultiHieron

Initialize:$\forall v\in Y:\mathbf{w}_0^v=\mathbf{0}, i=0$
1. **for** $i=1,2,...,m$**:**
Using: $(\mathbf{x}_i,\mathbf{y}_i)$
2. Predict: $\hat{\mathbf{y}}=\arg\max_{y\in Y}(|\mathbf{y}_i|)\sum_{v\in P(y)}\mathbf{w}_i^v\cdot\mathbf{x}_i$
3. Update:
$w_{i+1}^v=w_i^v+\alpha_i\mathbf{x}_i$**, if** $v\in\bigcup_{y\in\mathbf{y}_i}P(y)$
$w_{i+1}^v=w_i^v-\alpha_i\mathbf{x}_i$**, if** $v\in\bigcup_{\hat{y}\in\hat{\mathbf{y}}_i}P(\hat{y})$
$\alpha_i=\frac{L(\{\mathbf{w_i^v}\},\mathbf{x}_i,y_i)}{\gamma(\mathbf{y}_i,\hat{\mathbf{y}}_i)||\mathbf{x}_i||^2}$

Algorithm 2 is similar to algorithm 1. It iterates through every one of the m instances in the training set in line 1. During each iteration, line 2 gets $|\mathbf{y}_i|$ predictions, and 3 preforms weight updates if necessary. The main difference between the two algorithms in line 3, where we are performing updates on multiple labels. Our update rule contains a different path symmetric difference formula, defined as eq. 5. Also, we update each relevant prediction and true class.

4 Analysis of MultiHieron

MultiHieron is more general than the original. It can be trivially shown that MultiHieron will reduce to Hieron on any single label categorization problem. If $\mathbf{y}_i$ and $\mathbf{r}$ have maximum size of 1 for all i, then $\gamma(\mathbf{y}_i,\mathbf{r})=\gamma(y_i,r)$, the symmetric difference of the path to only two labels. All training, updates, and predictions will remain the same.

In [1], Dekel et al. provide a theorem that implied that the cumulative loss suffered by online Hieron is bounded as long as the margin requirements are satisfied. Proof follows from Theorem 1 in the aforementioned paper.

Theorem 4.1. *Let $\{(\mathbf{x}_i,\mathbf{y}_i)\}_{i=1}^m$ be a sequence of examples where $\mathbf{x}_i\in\mathbb{R}^n$ and $\mathbf{y}_i\in Y$. Assume there exists a set $\{\omega^v:\forall v\in y\}$ that satisfies the margin condition for all $1\leq i\leq m$. Then, the following holds,*

$$\sum_{i=1}^m L^2(\{\mathbf{w}_i^v\},\mathbf{x}_i,\mathbf{y}_i)\leq\sum_{v\in Y}||\omega^v||^2\gamma_{max}R^2 \tag{9}$$

where for all i, $||\mathbf{x}_i|| \leq R$ *and* $\gamma(\mathbf{y}_i, \hat{\mathbf{y}}_i) \leq \gamma_{max}$.

5 Experimentation

In experimentation, two datasets were used, the OntoNews corpus and the ASRS flight anomaly database. The ASRS flight anomaly database will be discussed in great detail in section 5.1. The OntoNews corpus is a collection of 290 ontology annotated news reports (see [5] and [4] for more information about this corpus). These were annotated using the Proton ontology. [7]

We convert each document to tf-idf weights and treat all annotations as labels for the document. For example, if a document discussed a football game, it would mention a Player, SportsBuilding, SportsTeam, etc., that would all be treated as labels for the document. The dataset was divided into 250 training documents and 40 test documents.

5.1 ASRS Anomaly Reports Dataset

The Aviation Safety Reporting System (ASRS) database is a repository of voluntary, confidential safety information provided by aviation personnel of all ranks, including pilots, controllers, mechanics, flight attendants and dispatchers. The database includes almost 150,000 incident reports submitted over more than 30 years. It has two major characteristics that distinguish it from other datasets: a document may have multiple anomalies (multi-categories/multi-labels) and each category belongs in a structured hierarchy. The difficulties associated with categorizing the documents as highly unstructured free form texts was addressed previously in [5], however, they lost a significant amount of highly relevant information by neglecting the underlying hierarchical structure of the categorization set. The ASRS database has 13 major classes, which are fairly general observations such as "inflight encounter" or "conflict", followed by 55 sub classes, which are much more specific. A report might be labeled as both "inflight encounter : weather" and "cabin event : passenger misconduct", if, say, it was a report about a passenger acting particularly angrily about the weather disturbing his flight.

In experimentation, a subset of the documents were chosen at random and tf-idf weights were generated to transform the document to vector forms. Then the feature space was determined through entropy calculations. Figure 2 shows the properties of the database. Our feature space has been restricted to 3814 distinct words, chosen by entropy analysis

5.2 Experimental setup

We used a **threshold based** method to test prediction accuracy. In both systems we pass an instance $\mathbf{x}_i$ and get back a list of all labels and their corresponding weights generated for that instance ($\mathbf{z}$). We then normalize these weights ($\hat{\mathbf{z}}$). Finally, we call our threshold, T, the point that immediately passes the sum of all true weights.

Training instances	20000
Testing instances	10000
Features per instance	3814
Minimum labels per instance	1
Maximum labels per instance	10
Average labels per instance	2.71
Maximum depth of hierarchy	3
Number of leaf nodes	55

Figure 2. ASRS selected dataset information

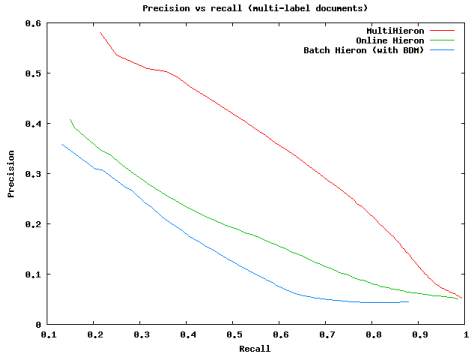
$$\sum_{p \in \mathbf{p}_t} p \leq T \tag{10}$$

5.3 Results

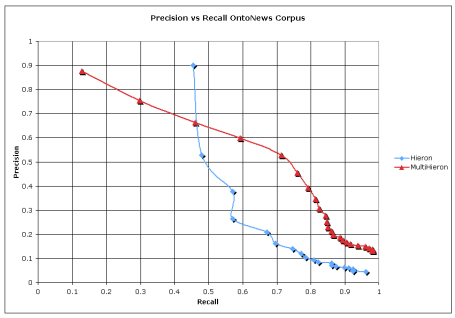
We vary the threshold, T, from 0 to 1 in 0.01 increments. T distinguishes positive predictions from negative predictions. We can use this to calculate a precision/recall curve, seen in figure 3(a). Lets examine the graph by looking at the points where recall is 0.6. Notice how the precision 0.35 for MultiHieron and 0.15 for Hieron. We can conclude that the MultiHieron does a significantly better job at classifying documents in this database that demonstrates superiority of our approach. Compare with figure 3(b), our base set, where a similar behaviour is exhibited.

Using a hierarchical error metric called BDM from [5], we generate an augmented precision versus recall curve. Figure 4 gives a dramatic lessening of the distance between the two curves. This is in part due to the flatness of the tree. This hierarchical measure gives error values in the range $[0, 1]$ instead of the binary values given in ordinary classification metrics. At an augmented recall of 0.6 in the curve, augmented precision is 0.3 and 0.35 for Hieron and MultiHieron, respectively. MultiHieron is more suited to the task of multiple label learning.

MultiHieron showed a speedup over Hieron. We observe that this speedup almost corresponds to the average 2.7 labels per document given in figure 2. Training the entire ASRS dataset with Hieron takes about 605.8s and training MultiHieron 253.7s. Training with multiple labels on the original Hieron algorithm was a multiple step process, with the majority of time spent calculating all of the possible prediction paths. With MultiHieron, only one prediction calculation need be made for each document, therefore the speedup is proportional to the average number of labels per document. Testing uses the exact same algorithms and data



(a) ASRS database



(b) OntoNews corpus

Figure 3. Precision versus recall for Hieron and MultiHieron

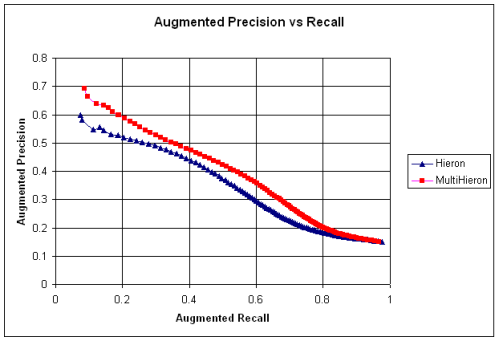


Figure 4. Augmented precision versus recall for Hieron and MultiHieron on ASRS database.

structures because the underlying prediction model for the algorithm was not changed, therefore testing speed is the same.

6 Conclusions and future work

We have presented a more generalized hierarchical perceptron algorithm, capable of doing multi-label document training. This new algorithm preserves all of the properties of original Hieron algorithm, and uses the same principles to preform multi-label learning.

We used ASRS and OntoNews datasets for experimentation. The ASRS flight anomaly database posed two unique problems. First, the anomaly reports could have more than one label. Second, labels belong to a structured hierarchy. We addressed these problems with the MultiHieron algorithm. MultiHieron showed improvement in both performance and accuracy over Hieron in multi-label learning over two data sets.

References

[1] O. Dekel, J. Keshet, and Y. Singer. Large margin hierarchical classification. In *ICML '04: Proceedings of the twenty-first International Conference on Machine Learning*, page 27, New York, NY, USA, 2004. ACM.

[2] S. Dill, N. Eiron, D. Gibson, D. Gruhl, R. Guha, A. Jhingran, T. Kanungo, S. Rajagopalan, A. Tomkins, J. A. Tomlin, and J. Y. Zien. Semtag and seeker: bootstrapping the semantic web via automated semantic annotation. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 178–186, New York, NY, USA, 2003. ACM.

[3] N. Guarino, C. Masolo, and G. Vetere. Ontoseek: Content-based access to the web. *IEEE Intelligent Systems*, 14(3):70–80, 1999.

[4] Y. Li and K. Bontcheva. Hierarchical, perceptron-like learning for ontology-based information extraction. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 777–786, New York, NY, USA, 2007. ACM.

[5] D. Maynard, W. Peters, and Y. Li. Metrics for evaluation of ontology-based information extraction. *WWW 2006 Workshop on "Evaluation of Ontologies for the Web" (EON)*, 2006.

[6] NASA. Aviation safety reporting system database. http://asrs.arc.nasa.gov/search/database.html.

[7] S. Project. Proto ontology (proton). http://proton.semanticweb.org.